Problem1:

For this question, we will calculate the expected value and standard deviation of the 3 different methods based on the formulas from the class handouts, as shown below:

1. Classical Brownian Motion

$$P_t = P_{t-1} + r_t$$

2. Arithmetic Return System

$$P_t = P_{t-1}\left(1 + r_t\right)$$

3. Log Return or Geometric Brownian Motion

$$P_t = P_{t-1}e^{r_t}$$

First, we need to manually calculate for each method as shown below:

$r_t \sim N(0, \delta^2 = 0.01)$.

$\Rightarrow E[r_t] = 0. \quad \delta = \sqrt{0.01} = 0.1$

① Classical Brownian Motion:

$P_t = P_{t-1} + r_t$

$\therefore E[P_t] = E[P_{t-1} + r_t]$

$= E[P_{t-1}] + E[r_t]$

$= E[P_{t-1}] + 0 = E[P_{t-1}] = P_{t-1}$

$Var[P_t] = Var[P_{t-1} + r_t]$

$= Var[P_{t-1}] + Var[r_t]$

$= Var[P_{t-1}] + 0.01 = 0.01$

$\therefore$ Here $E[P_t] = P_{t-1}$, Std $= 0.1$

② Arithmetic Return: $P_t = P_{t-1}(1+r_t)$

$E[P_t] = E[P_{t-1}(1+r_t)] = P_{t-1} * E[(1+r_t)] = (P_{t-1}) \times (E[r_t]+1) = P_{t-1}$
$= P_{t-1}$

$Var[P_t] = Var[P_{t-1}(1+r_t)] = P_{t-1}^2 * Var[(1+r_t)] = P_{t-1}^2 * Var(r_t) = 0.01 P_{t-1}^2$

thus, $E[P_t] = P_{t-1}$, std $= 0.1 P_{t-1}$

③. Geometric Brownian Motion: $P_t = P_{t-1} e^{r_t}$

$\therefore E[P_t] = E[P_{t-1} e^{r_t}] = P_{t-1} E[e^{r_t}] = P_{t-1} * e^{\frac{\delta_t^2}{2} + \mu} = P_{t-1} * e^{\frac{\delta_t^2}{2}} = e^{0.005} P_{t-1}$

$Var[P_t] = Var[P_{t-1} e^{r_t}] = P_{t-1}^2 Var[e^{r_t}] = P_{t-1}^2 e^{\delta_t^2} (e^{\delta_t^2} - 1)$

$= P_{t-1}^2 e^{0.01} \times (e^{0.01} - 1)$.

Thus, $E[P_t] = e^{0.005} P_{t-1}$

std $= \sqrt{P_{t-1}^2 e^{0.01} \times (e^{0.01} - 1)} = P_{t-1} * \sqrt{e^{0.01} \times (e^{0.01} - 1)}$

For the second part, we predefined the Pt-1 = 100 and use n = 100000 to simulate the numbers for those three methods, and the results based on the manually calculated formulas should be:

Mean:

Classical Brownian Motion = Pt-1 = 100

Arithmetic Return System = Pt-1 = 100

Geometric Brownian Motion = e^(0.005)*Pt-1 = 100*e^(0.005) = 100.50125

Standard Deviation:

Classical Brownian Motion = 0.1

Arithmetic Return System = 0.1*Pt-1 = 0.1*100 = 10

Geometric Brownian Motion = sqrt(e^(0.01)*(e^(0.01)-1))*Pt-1 = 10.0753

We also generate the simulated results from the code and get the following outputs:

```
                                  Mean    Std Dev
Classical Brownian Motion    99.999680   0.099858
Arithmetic Return System     99.968008   9.985796
Geometric Brownian Motion   100.467687  10.057791
```

As you can see from the results above, the simulated and expected numbers are pretty similar, meaning that we calculated the numbers correctly.

Problem2:

For this part, we're going to use the arithmetic return system to calculate the returns

2. Arithmetic Return System

$$P_t = P_{t-1}\left(1 + r_t\right)$$

Then, we're going to modify the jl code from the in-class example to write the return_calculate() function in python, in the code we're going to use the discrete method and especially this part of code:

```python
if method.upper() == "DISCRETE":
    p2 = (p[1:, :] / p[:-1, :]) - 1.0
elif method.upper() == "LOG":
    p2 = np.log(p[1:, :] / p[:-1, :])
else:
    raise ValueError(f"Invalid method '{method}'. Must be 'DISCRETE' or 'LOG'.")
```

to get the returns, which is shown as below:

	Date	SPY	AAPL	MSFT	AMZN	TSLA	GOOGL	GOOG	META	NVDA	BRK-B	JPM	JNJ	UNH	
0	2023-09-06 00:00:00	-0.006723	-0.035793	-0.002009	-0.013914	-0.017817	-0.009649	-0.009802	-0.003265	-0.030550	0.003329	-0.001653	-0.016617	-0.008049	-0.0
1	2023-09-07 00:00:00	-0.003070	-0.029249	-0.008922	0.018395	-0.001707	0.005950	0.006131	-0.001671	-0.017424	0.000359	-0.008554	0.012784	0.016501	0.0
2	2023-09-08 00:00:00	0.001506	0.003492	0.013216	0.002757	-0.011889	0.008280	0.007342	-0.002612	-0.014468	0.003731	0.000765	0.003312	-0.004473	0.0
3	2023-09-11 00:00:00	0.006577	0.006623	0.010979	0.035231	0.100925	0.003960	0.003936	0.032462	-0.008646	0.006526	0.004380	0.013079	-0.002891	-0.0
4	2023-09-12 00:00:00	-0.005486	-0.017061	-0.018258	-0.013068	-0.022297	-0.011540	-0.012124	-0.019183	-0.006818	0.006183	0.013014	0.005656	0.001085	-0.0

And the table includes 249 rows and 100 columns, then we need to remove the META ones

Then using the five different methods to calculate the results is shown as below:

```
VaR using Normal Distribution: -0.03817295454890715
VaR using EWMA: -0.0026055505180793O4
VaR using MLE-Fitted T Distribution: -0.03242585900409048
VaR using AR(1) Model (ARIMA): -0.03812381631443842 5
VaR using Historical Simulation: -0.029464221305067637
```

As you can see from the results, the outputs from each method are different, EWMA has the largest number, and Normal Distribution has the smallest number of VaR, and because of the logic behind the exponentially weighted method, indicating that recent market conditions are pretty stable. Also from the results, we can see that a combination of different methods should be more suitable for a real case.

Problem3:

In this problem, I used both Arithmetic Return system and log return for returns then used both exponentially weighted covariance to get the VaR. I also used the latest stock price in the DailyPrices sheet to calculate the filtered portfolio value for A, B, C and total. Arithmetic Return method is good for short-term analysis cause it assumes the price changes are linear over time, and log return method is more suitable for longer term analysis because the returns are compounded over time.

The results I got is shown as below:

```
Portfolio A EWMA VaR in $ (Arithmetic Returns): 17565.281020663795
Portfolio A EWMA VaR in $ (Log Returns): 17623.71789295673
Portfolio B EWMA VaR in $ (Arithmetic Returns): 9391.747566238244
Portfolio B EWMA VaR in $ (Log Returns): 9418.442950899931
Portfolio C EWMA VaR in $ (Arithmetic Returns): 19072.13391550189
Portfolio C EWMA VaR in $ (Log Returns): 19179.58369077331
Total Portfolio EWMA VaR in $ (Arithmetic Returns): 44708.420728849786
Total Portfolio EWMA VaR in $ (Log Returns): 44880.291275235126
```

As you can see from the screenshot, The VaR value in $ amount for log return method is always slightly higher than the arithmetic returns, meaning that log return method can provide a more conservative analysis for the risk because of the compounding effects. Besides that, we can see that the total portfolio VaR numbers is less than the direct sum for the 3 portfolios, showing that there're some diversifications within the portfolio to reduce the risks.